**Как создать слайдер экранов с использованием ViewPager2 на Kotlin**

Реализуем андроид-приложение, в котором можно будет листать экраны свайпом вправо или влево. Для реализации слайдера будем использовать компонент ViewPager2 из библиотеки androidx. На каждом экране будет его порядковый номер, а листать их можно будет бесконечно (теоретически).

ViewPager2 – это не просто усовершенствованный ViewPager, ViewPager2 под капотом использует компонент для работы со списками RecyclerView, и это дает больше удобства и преимуществ, например, в построении динамических вкладок на основе пагинируемых списков, и многое другое.

**Добавляем ViewPager2 в проект Android Studio**

Чтобы использовать ViewPager2, необходимо в файл сборки модуля App вашего проекта добавить зависимости библиотек материальных компонентов и ViewPager2.



|  |  |
| --- | --- |
| 1  2 | implementation ("androidx.viewpager2:viewpager2:1.0.0") |

Далее откроем файл разметки res/layout/activity\_main.xml, где вместо TextView разместим компонент ViewPager2:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"      xmlns:app="http://schemas.android.com/apk/res-auto"      xmlns:tools="http://schemas.android.com/tools"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent"      tools:context=".MainActivity">        <androidx.viewpager2.widget.ViewPager2          android:id="@+id/pager"          android:layout\_width="match\_parent"          android:layout\_height="match\_parent"          app:layout\_constraintBottom\_toBottomOf="parent"          app:layout\_constraintLeft\_toLeftOf="parent"          app:layout\_constraintRight\_toRightOf="parent"          app:layout\_constraintTop\_toTopOf="parent" />    </androidx.constraintlayout.widget.ConstraintLayout> |

Удалим вывод текста, размеры компонента зададим по родителю и укажем идентификатор.

**Фрагмент для слайдера экранов**

В главном пакете приложения создадим фрагмент, щелкнув правой кнопкой мыши по имени пакета и в контекстном меню выбрав команду New> Fragment> NumberFragment

Сначала в папке res/layout отредактируем его макет:



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | <?xml version="1.0" encoding="utf-8"?>  <androidx.constraintlayout.widget.ConstraintLayout      xmlns:android="http://schemas.android.com/apk/res/android"      xmlns:app="http://schemas.android.com/apk/res-auto"      xmlns:tools="http://schemas.android.com/tools"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent"      tools:context=".NumberFragment">          <TextView          android:id="@+id/textView"          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          android:text="1"          android:textSize="254sp"          app:layout\_constraintBottom\_toBottomOf="parent"          app:layout\_constraintEnd\_toEndOf="parent"          app:layout\_constraintStart\_toStartOf="parent"          app:layout\_constraintTop\_toTopOf="parent" />    </androidx.constraintlayout.widget.ConstraintLayout> |

Изменим корневой компонент на ConstraintLayout, а компоненту TextView зададим идентификатор, размер текста побольше и расположение по центру экрана.

Теперь отредактируем класс фрагмента:



|  |  |
| --- | --- |
|  | const val ARG\_OBJECT = "object"    class NumberFragment : Fragment() {        override fun onCreateView(          inflater: LayoutInflater, container: ViewGroup?,          savedInstanceState: Bundle?      ): View? {          // Inflate the layout for this fragment          return inflater.inflate(R.layout.fragment\_blank, container, false)      }        override fun onViewCreated(view: View, savedInstanceState: Bundle?) {          arguments?.takeIf { it.containsKey(ARG\_OBJECT) }?.apply {              val textView: TextView = view.findViewById(R.id.textView)              textView.text = getInt(ARG\_OBJECT).toString()          }      }    } |

Этот фрагмент будет использоваться для каждого нового экрана в приложении, мы будем создавать новый его экземпляр и передавать туда его порядковый номер.

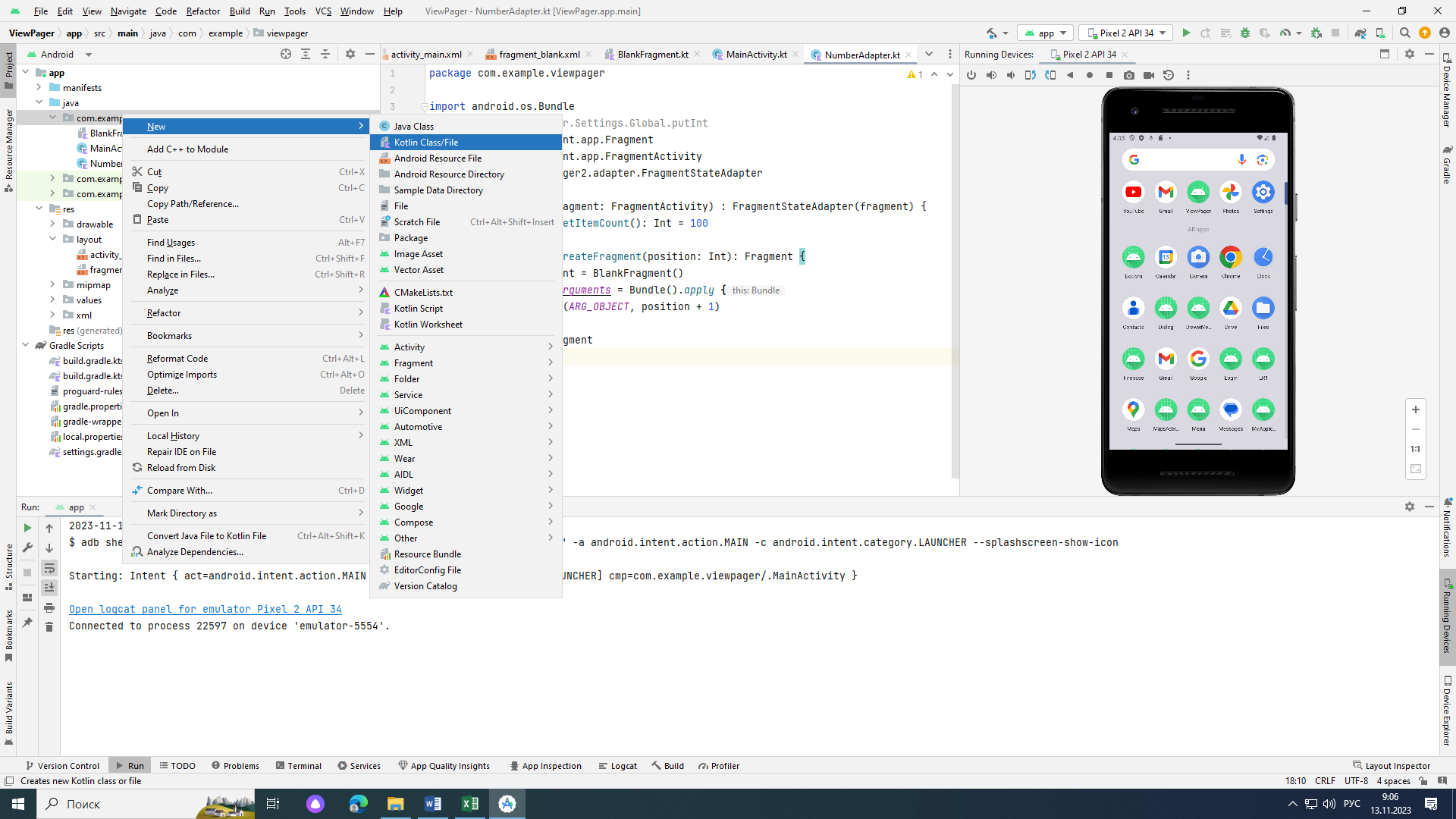
Вне тела класса создаем константу для ключа аргумента, который будем передавать в каждый новый экземпляр фрагмента.

В переопределенном методе onViewCreated получаем аргумент, находим текстовое поле и передаем туда значение аргумента для отображения.

**Адаптер для ViewPager2**

Создадим адаптер, задачей которого будет предоставление фрагментов для слайдера:

Создаем новый класс с именем NumberAdapter



и модифицируем его содерджание



|  |  |
| --- | --- |
|  | class NumberAdapter(fragment: FragmentActivity) : FragmentStateAdapter(fragment) {      override fun getItemCount(): Int = 100        override fun createFragment(position: Int): Fragment {          val fragment = NumberFragment()          fragment.arguments = Bundle().apply {              putInt(ARG\_OBJECT, position + 1)          }          return fragment      }    } |

При создании адаптера наследуем его от FragmentStateAdapter, который, в свою очередь, унаследован от RecyclerView.Adapter, а это значит, что мы можем использовать все возможности RecyclerView.Adapter для работы с ViewPager2.

Как понятно из названия, FragmentStateAdapter работает с фрагментами. В зависимости от релизации, в конструктор адаптера нужно передать экземпляр или класса Fragment, или класса FragmentActivity. Это нужно для синхронизации с жизненным циклом фрагментов слайдера. Передадим экземпляр класса FragmentActivity, от которого мы потом унаследуем MainActivity текущего проекта.

Требуется переопределить метод getItemCount, возвращающий общее количество элементов списка. Мы ограничим количество элементов сотней, просто для этого примера.

Переопределим метод createFragment, возвращающий фрагмент для каждого элемента слайдера. Здесь создаем экземпляр нашего фрагмента и передаем ему в качестве аргумента порядковый номер его позиции, а поскольку нумерация начинается с нуля, прибавляем единицу.

**MainActivity**

Осталось завершить реализацию в MainActivity:



|  |  |
| --- | --- |
|  | class MainActivity : FragmentActivity() {        private lateinit var adapter: NumberAdapter      private lateinit var viewPager: ViewPager2        override fun onCreate(savedInstanceState: Bundle?) {          super.onCreate(savedInstanceState)          setContentView(R.layout.activity\_main)            adapter = NumberAdapter(this)          viewPager = findViewById(R.id.pager)          viewPager.adapter = adapter      }  } |

Унаследуем его от FragmentActivity. Объявим переменные для адаптера и вьюпейджера.

В методе onCreate инициализируем адаптер, передав ему текущее активити как владелец жизненного цикла, находим вьюпейджер по идентификатору и передаем ему адаптер. Все.

Запустите приложение на эмуляторе или смартфоне. Сразу появится экран с номером 1, а при свайпе справа на лево будут открываться экраны с последующими номерами. Обратным свайпом можно вернуться назад.